Multimodal 3D Hand Pose Enhancement for Sign Language

Álvaro Budria
alvaro.francesc.budria@estudiantat.upc.edu

Laia Tarrés
laia.tarres@upc.edu

Xavier Giró
xavier.giro@upc.edu

Abstract

The application of recent deep learning breakthroughs to the domain of sign language has yielded very promising results. However, sign language processing systems depend on large amounts of labeled high-quality data to work properly. Current hand pose estimation methods are often unreliable and do not always produce estimations with enough quality. To mitigate this issue, we explore the applicability of the novel Body2Hands method for the obtainment of high-quality hand pose estimations.

1. Introduction

It is estimated that for over 460 million people worldwide, sign languages are the primary means of communication. Although sign languages are used by millions of people everyday to communicate, the vast majority of communications technologies nowadays are designed to support spoken or written language, but not sign languages.

There have been recent works in sign language processing [1, 7, 15, 18] showing that modern machine learning architectures can help break down these barriers for sign language users.

However, these systems depend on the existence of large volumes of labeled high-quality sign language data for achieving their objectives. Directly processing raw sign language video data is too time-consuming and resource expensive to be reasonably feasible in most settings. For this reason, instead of dealing with raw video data, one usually extracts the keypoints (i.e. the relevant body points, face landmarks and other bodily features) from the video frames. This keypoint representation makes systems more robust to changes in background and variability among signers. Moreover, the standard procedure for obtaining the keypoint representation of sign language speakers is based on pose detection systems such as OpenPose [2] and MediaPipe [8], which are often unreliable and tend to introduce noise into the data.

This motivates the introduction of systems for postprocessing pose detectors' estimations. In this work we explore the applicability of the novel Body2Hands [9] method on the domain of sign language, for the obtainment of higher quality hand pose estimations, with the aim of improving upon the current pose extraction systems.

2. Related Work

2.1. Body2Hands

Body2Hands [9] is a recently proposed method for generating convincing conversational hand gesture given the speaker's arms. More precisely, given a sequence of body poses in rotational form $\mathbf{B} = \{b_t\}_{t=1}^{T}$, Body2Hands produces a sequence of hand poses $\mathbf{H} = \{h_t\}_{t=1}^{T}$:

$$\mathbf{H} = \mathcal{G}(\mathbf{B}). \tag{1}$$

2.1.1 Setup and Model Architecture

Body2Hands adopts a generative adversarial framework for generating the hand motion.

The generator $\mathcal{G}$ is a fully-convolutional 1D encoder-decoder network. More precisely, the model follows a U-Net [12] architecture, with two layers at the encoder and two at the decoder. The task of generating hand poses is set as a regression task via an L_1 loss:

$$\mathcal{L}_{L_1} = \|\hat{\mathbf{H}} - \mathcal{G}(\mathbf{B})\|_1. \tag{2}$$

To produce more realistic movement and avoid blending different modes of motion, the authors introduce an adversarial discriminator $\mathcal{D}$. They also introduce a function Δ to compute the deltas between consecutive poses (i.e. $[h_2 - h_1, ..., h_T - h_{T-1}]$). The discriminator $\mathcal{D}$ maximizes the following objective while the generator $\mathcal{G}$ minimizes it:

$$\mathcal{L}_{GAN}(\mathcal{G}, \mathcal{D}) = \mathbb{E}_{\mathbf{H}}[\log \mathcal{D}(\Delta(\mathbf{H}))] + \mathbb{E}_{\mathbf{B}}[\log 1 - \Delta(\mathbf{B})]. \tag{3}$$

The full objective is thus

$$\min_{\mathcal{G}} \max_{\mathcal{D}} \mathcal{L}_{GAN}(\mathcal{G}, \mathcal{D}) + \lambda \mathcal{L}_{L_1}(\mathcal{G}). \tag{4}$$

2.1.2 Leveraging Body Pose Priors

As presented in [9], this GAN can be extended by using additional hand images as input to the model:

$$\mathbf{H} = \mathcal{K}(\mathbf{I}_h, \mathbf{B}), \tag{5}$$

where $\mathbf{I}_h$ is a series of hand images cropped around the left and right hand regions of the input video. For our experiments, we extracted hand crops from the How2Sign dataset and then utilized a ResNet-50 as feature extractor, following the procedure described in Body2Hands.

2.1.3 Leveraging Text Priors

We modified the architecture of the model to also accept a single vector embedding as input:

$$\mathbf{H} = \mathcal{K}(\mathbf{T}, \mathbf{B}), \tag{6}$$

where $\mathbf{T}$ is a vector embedding summarizing the text associated with sequences $\mathbf{B}$ and $\mathbf{H}$. In our experiments, we leveraged text embeddings obtained with the text encoder of the novel CLIP model [11].

2.2. How2Sign Dataset

How2Sign [4] is a large-scale collection of multimodal and multiview sign language videos in American Sign Language (ASL) for over 2500 instructional videos selected from the existing How2 dataset [14].

How2Sign consists of more than 80 hours of American Sign Language videos, with sentence-level alignment for more than 35k sentences. It features a vocabulary of 16k English words that represent more than two thousand instructional videos from a broad range of categories. The dataset comes with a rich set of annotations including category labels, text annotations, as well automatically extracted 2D keypoints for more than 6M frames.

We made extensive use of the How2Sign dataset, and chose it for all of our experiments, due to the quality of its pose estimations, its extensive transcriptions to English, and its category annotations.

3. Skeletal Model and Data Representation

A key aspect that makes sign language processing a difficult endeavor, is gathering and representing sign language data. Here we present the data representations we have used. We focus specifically on how to represent the body/skeleton of a speaker.

The How2Sign dataset [4] provides videos along with the 2-dimensional coordinates of the speakers' body joints or keypoints, which were obtained with OpenPose [2]. Therefore we can directly use these 2D keypoints to represent the joints of the speaker's skeleton. In this case, we would represent the speaker's body at a certain time frame t as a vector of coordinates of the form $x_t = (x_0, y_0, x_1, y_1, ..., x_{49}, y_{49})$. The speaker's kinematic tree has a total of 50 keypoints (21 for each hand, 3 for each arm, and 2 for the neck).

This 2D representation, however, is not rich enough, and not quite suitable for sign language, where occlusions and changes in angle are frequent. To alleviate this issue, we made use of a lifting method specifically designed for sign language processing which converts 2D keypoints to sensible 3D representations.

3.1. Lifting From 2D to 3D

In [17] a method was proposed for lifting sequences of 2D sign-language keypoints to 3D coordinates. This method makes use of several invariants in the human body to obtain quality 3D representations. We changed the prespecified kinematic tree to our own custom one, adapting it to the data from the How2Sign dataset.

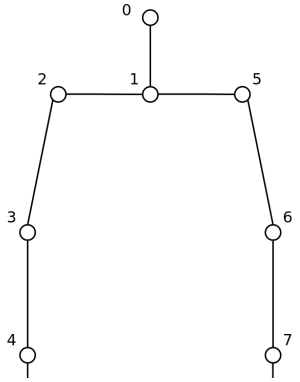

Figure 1. Kinematic structure. Body view.

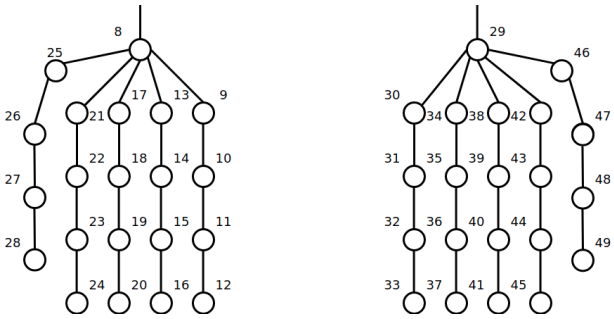

Figure 2. Kinematic structure. Hands view.

3.2. From Cartesian to Rotational Representation

Although this 3D Cartesian representation allows handling occlusions and different camera angles much more effectively, it suffers from sensitivity to scale and length of the speaker's limbs. To obtain a representation that is invariant to changes in scale, we decided to follow [9] and [16], by converting the Cartesian coordinates to a 6D rotational representation [19] called R6D. In essence, with a rotational

representation such as axis-angle or R6D, we represent a body joint as its rotation with respect to its parent joint.

The conversion process involves a number of steps described in Algorithm 1. Since we compute the rotation of a joint against its parent, we perform the conversion by traversing a kinematic tree (ours is described in Figures 1 and 2). It is necessary to define a root bone, from which to start the traversal. We set the root bone to be the neck, since it is a very non-informative body part of a signer and it is therefore safe to assume it remains fixed.

For each triplet of joints, we compute vectors $\vec{u}$ and $\vec{v}$, representing the parent bone and the "rotated" one, respectively. Then, we obtain the axis $\hat{a}$ and angle θ by which the rotation from the parent to the child joints is achieved. Vector $\theta\hat{a}$ is the *axis-angle* representation of the central joint w.r.t to its parent joint. From this, we can easily obtain its rotation matrix and its R6D representation (which corresponds to vectorizing the first two columns of the rotation matrix).

3.3. From Rotational Representation to Cartesian

We can easily invert this process to obtain Cartesian coordinates from an axis-angle representation. However, we need to know beforehand the kinematic tree, the root bone, and the bone length that were used for the initial conversion. If this is known, we can leverage the parent bone of the current joint and Rodrigues' rotation formula [10] to reconstruct the position of the next joint in the kinematic tree, given its parent and its rotation. The full process is described in Algorithm 2.

Algorithm 1 Conversion from Cartesian 3D to axis-angle representaton

Require: K ▷ the kinematic structure
Require: xyz ▷ a vector $(x_1, y_1, z_1, ..., x_n, y_n, z_n)$
$aa \leftarrow \{\}$
for iBone in K **do**
  $id_J, id_E, id_B \leftarrow K[iBone]$
  ▷ A bone has J and E as joints. B is J's parent
  $J \leftarrow xyz[id_J * 3 : id_J * 3 + 3]$
  $E \leftarrow xyz[id_E * 3 : id_E * 3 + 3]$
  $B \leftarrow xyz[id_B * 3 : id_B * 3 + 3]$
  $\vec{u} \leftarrow J - B$
  $\vec{v} \leftarrow E - J$
  $\theta \leftarrow \arccos \frac{\vec{u} \cdot \vec{v}}{\|u\|\|v\|}$ ▷ rotation angle
  $\hat{a} \leftarrow \frac{\vec{u} \times \vec{v}}{\|u\|\|\vec{v}\|}$ ▷ axis of rotation
  append $\theta\hat{a}$ to aa
end for
return aa

Algorithm 2 Conversion from axis-angle to Cartesian 3D representaton

Require: K ▷ the kinematic structure
Require: aa ▷ a vector $(x_1, y_1, z_1, ..., x_n, y_n, z_n)$
Require: J, B ▷ joints belonging to root bone
Require: L ▷ bone lengths
$xyz \leftarrow \{\}$
append J, B to xyz ▷ root is already in 3D
$i \leftarrow 0$
for iBone in K **do**
  $id_J, id_E, id_B \leftarrow K[iBone]$
  $J \leftarrow xyz[id_J * 3 : id_J * 3 + 3]$
  $B \leftarrow xyz[id_B * 3 : id_B * 3 + 3]$
  $\vec{u} \leftarrow \frac{J-B}{\|J-B\|}$
  $\theta \leftarrow \|aa[i * 3 : i * 3 + 3]\|$
  $\hat{a} \leftarrow \frac{1}{\theta} \cdot aa[i * 3 : i * 3 + 3]$
  $i \leftarrow i + 1$
  ▷ Rodrigues' formula
  $\vec{v} \leftarrow \vec{u}\cos\theta + (\hat{a} \times \vec{u})\sin\theta + \hat{a}(\hat{a} \cdot \hat{u})(1 - \cos\theta)$
  ▷ retrieve the next joint from the current one
  $E \leftarrow J + L[iBone]\vec{v}$
  append E to xyz
end for
return xyz

4. Experimental Results

4.1. Transferring Body2Hands to Sign Language

We were interested in assessing whether off-the-shelf methods from other domains may be useful in the domain of sign language. In particular, we tested the method called Body2Hands [9], which was initially designed for generating hand poses in conversational settings. We directly applied it on the domain of sign language. Without modifying the model's architecture in any way, we trained it on the How2Sign dataset (after having previously converted the data from 2D to R6D).

4.1.1 Conditioning on Text

The How2Sign dataset contains text annotations at sentence level for each of the clips. Note the text is not completely aligned with each of the signer's poses. However, the correspondance between a clip and its textual sentence is available in the dataset. We leverage this information to condition the model on the sentence-level textual information.

The initial Body2Hands model allows conditioning on image embeddings by concatenating T image embeddings (one per frame) to the T pose embeddings, along dimension Q of the pose embeddings. This is not suitable for text, as we do not posses information at the frame level. To alleviate this issue, we adapted the model architecture to

input	train	val	test
body pose	2.36	2.38	2.39
body pose + text embedding	2.37	2.38	2.38
body pose + image crops	2.30	2.39	2.37

Table 1. L_1 error obtained by each model configuration.

be able to receive textual information as input. Instead of concatenating the features to the input of the model's generator, we concatenate the sentence embedding to the latent representation produced at the bottleneck of the generator's encoder-decoder structure.

Regarding the text embeddings, we obtained them by feeding the sentence-level textual information to a pretrained text encoder. In particular, we used the text encoder of the novel CLIP [11] model, which generates a vector of size 512 for each given input sentence.

4.1.2 Conditioning on Images

We also leverage the capability of the Body2Hands model to receive image features as input. No modification to the model's architecture was needed for this.

Hand crops of the signer's left and right hands were extracted, thus obtaining $2 \times T$ crops per clip. To generate the image feature representations, we used a ResNet-50 [5] network pretrained on ImageNet [3]. For each frame in a clip, we extracted a vector feature representation, thus obtaining $2 \times T$ vectors per clip. The Body2Hands model then reduces their dimensionality so that later they can be concatenated along the pose embeddings before entering the generator module.

4.1.3 Experiments

We assess whether the model can be successfully applied to sign language data by training and testing it on the How2Sign dataset. We also check the impact that the hand prior and the text prior may have on the model's generative power.

We thus train the model on three different settings, namely, with body pose only as input, with body pose plus text embedding and with body pose plus image embeddings. We use a batch size of 256, with Adam as optimizer and with a learning rate of 10^{-4}. As described in [9], we train with an adversarial loss every third epoch, and without, for all other epochs. Training time is 2 hours on a single GPU for 200 epochs.

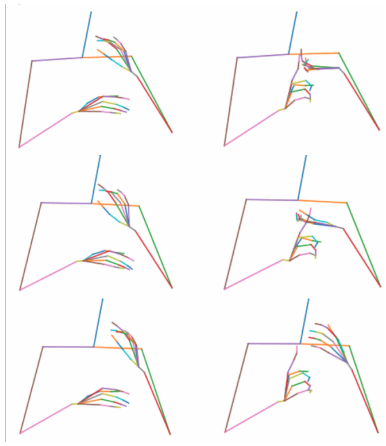

Figure 3. Left: hand poses generated with off-the-shelf Body2Hands model. Right: original poses from How2Sign dataset.

Table 1 reports the reconstruction error (i.e. the L_1 distance) between the generated hand pose and the GT pose. Notice that the error on the train set is very similar to that obtained on the test and validation sets. Figure 3 shows how the model is not generating any meaningful gestures. Rather, it suffers from a common phenomenon in GANs called mode collapse, which is due mainly to a lack of representation power in the generative model.

Neither conditioning on the text nor on the hand crops yields improved results, therefore a different architecture will be needed for the model to work.

4.2. Assessing generative power through unmasking task

Since directly transferring the Body2Hands model to the sign language domain did not yield very good results, we became interested in evaluating the model's capacity, in order to determine how difficult the task at hand is and what kind of model may be adequate to solve it. More precisely, we were interested in knowing at which point the model falls into model collapse, that is, at which point the model stops generating a rich variety of poses and starts permanently generating the median pose.

Our approach was characterized by using as input not only the arms, but also the hands, while also masking some of the fingers. The model then must reconstruct the masked finger(s) from the arms and the non-masked fingers.

4.2.1 Experiments

We experimented with masking a different amount of fingers, from 1 to 5. Table 2 shows the reconstruction error, as well as the error per reconstructed finger, against the number of masked fingers. Although the error per masked finger decreases as we increase the number of masked fingers, the

# masked fingers	val error	(val error) / (masked finger)	(test error)	(test error) / (masked finger)
1	0.320	0.320	0.324	0.324
2	0.331	0.166	0.330	0.165
3	0.338	0.113	0.341	0.114
4	0.382	0.096	0.381	0.096
5	0.418	0.084	0.411	0.082

Table 2. L_1 reconstruction error against number of masked fingers.

overall quality of the reconstruction becomes worse, with fingers being less expressive and mobile. In Figure 4 we see how masking a single finger results in a reconstruction that is indistinguishable from the ground truth pose. Figure 5 contrasts with that: masking a whole hand results in a reconstruction that is inexpressive and inadequate in the context of sign language processing.

We did not find a specific number of masked fingers from which the model falls into permanently predicting the median pose; on the contrary, it seems that there is a gradual decrease in the quality of the reconstructed fingers as the number of masked ones increases. It must be noted, however, that those fingers that are not masked, are reasonably reconstructed by the model. So while the model has a hard time extrapolating fingers from the rest of the speaker's arm sand hands, it can nonetheless recover the non-masked inputs, and even remove some of the noise present in the inputs.

The conclusion is that the Body2Hands model does not have enough capacity to deal with the task of generating sign language poses. Nevertheless, the fact that the model can reconstruct one or two fingers with quality indicates that the model could potentially be used for enhancing sequences and removing noise that might be present in them. We explore this approach in the next section.

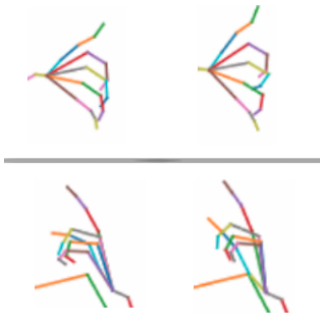

Figure 4. Two samples of reconstructed poses, having previously masked just 1 finger. On the left, the hand with the reconstructed finger; on the right, the ground truth. The model can adequately reconstruct a finger, given its context.

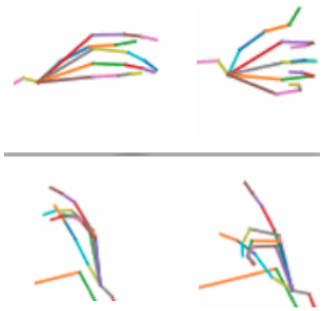

Figure 5. Two samples of reconstructed poses, having previously masked 5 fingers (a whole hand). On the left, the reconstructed hand; on the right, the ground truth. Notice how the model does not have enough capacity to reconstructed an entire masked hand.

4.3. Hand Pose Enhancement

A severe problem that hinders the performance of sign language processing systems is the difficulty of obtaining clean, usable data. The main sources of noise are the pose detection algorithms (e.g. OpenPose [2]) and the mechanisms for lifting data to 3D coordinates. In Figure 7, on the right, we can see an example of a non-realistic hand-shape that was produced after lifting it to 3D. Examples such as this one are bound to have a negative impact on a model's performance. This motivates the necessity of devising systems that allow to remove noise and inconsistencies in the poses.

In light of the results obtained with the masking experiments presented in the previous section, we decided to carry the idea further by having the model reconstruct both hands entirely. Our hypotheses is that the model acts as a regularizer, keeping only the most important features in the data, while discarding the noise.

Initially, we carried out a qualitative evaluation of our procedure, since reconstruction error is not an adequate measure in this case (due to the fact that we do not intend to reconstruct the poses, but to clean them).

We experimented with different inputs and outputs (see below for more details). In general, the model manages to filter out the most prominent noise, producing smoother hand poses. However, it presents the drawback that it also

tends to erase significant hand gestures that are relevant for sign language.

After a qualitative evaluation of the results, we decided that using the hand poses both as input and as output yields the best results in terms of noise removal.

To obtain a more quantitative evaluation of our approach, we decided to evaluate it against a topic detection task, which we present in the next section. For this section, we limit ourselves to presenting a more qualitative assessment of the enhancement mechanism under L_1 reconstruction loss.

4.3.1 Experiments

We experimented with utilizing different kinds of inputs for filtering out the noise in the hand poses. Thus, we tried utilizing as input to our model several other sources of data, besides the hand poses, namely, the arm poses, the hand crop images, and both at the same time.

We begin by exploring the possibility of enhancing the hand poses with the help of the body poses, thus having arm and hand poses as input, and just hand poses as outputs. Figure 6 shows an example of an improved sign language sequence. Notice that obviously wrong and noisy poses, such as those of the right hand in the 1st and 3rd frames, are fairly cleaner in the post-processed sequence.

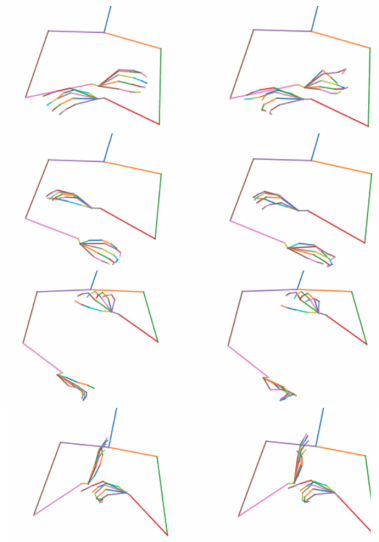

Figure 6. Left: enhanced poses. Right: original poses
For this sequence, body poses and hand poses were used as input.

In line with the results obtained during our masking experiments presented in the previous section, we found no improvement in terms of quality of the outputs when including also as input the image hand crop features. We can conclude that the model at hand does not make appropriate use of the visual features that it may receive and input and therefore a different architecture is needed for leveraging those features.

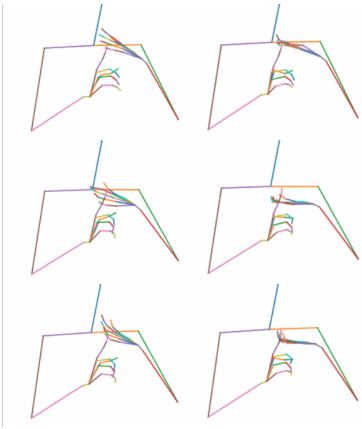

Figure 7. Left: enhanced poses. Right: original poses
For this sequence, only hand poses were used as input.

On the other hand, with hands only as input the results were ever so slightly better, compared to using multiple inputs. Figure 7 shows a sequence post-processed in this way. We hypothesize that by using additional inputs such as the arms or the hand crop image features, we are exceeding the capacity of the model, and that by limiting ourselves to the hand poses, we make more efficient use of the model's learning power.

5. Topic Detection

As mentioned above, the motivation for tackling the topic detection task on the How2Sign dataset is to provide a surrogate task by which to evaluate whether the hand pose enhancement method described in the previous section is an effective strategy. This task allows us to provide a more quantitative evaluation of the results.

There are 9 different categories in the data, and there is a notable class imbalance, with the most frequent class accounting for over 20% of the samples in the validation set. While we have not applied any resampling techniques to address this issue, it is necessary to take it into account when interpreting the models' performance.

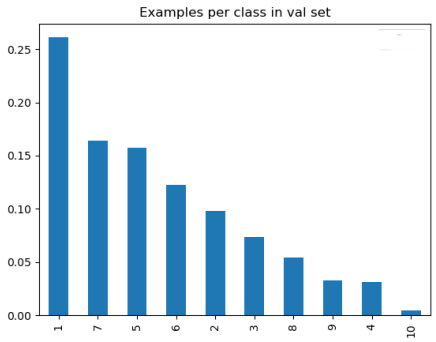

Figure 8. Bar chart representing the proportion each class represents in the validation set. There is a noticeable class imbalance.

By default, data in the dataset is available at *sentence-level*, that is, each sequence of keypoints $\mathbf{X} = \{x_t\}_{t=1}^{T}$ (with $x_t = (x_1, y_1, x_2, y_2, ..., x_{32}, y_{32})$) has a textual sentence associated. There are 31128 such sentence-level sequences for training, 1741 for validation, and 2322 for testing. However, we can group sentence-level sequences based on the original video they were extracted from, which results in 2192 sequences for training, 115 for validation and 149 for testing.

At first glance, sentence-level information seemed too limited for being classified. For instance, a sentence such as "I'm not going to use a lot, I'm going to use very very little." could very well be categorized both as belonging to "Games" and to "Food and Drinks". A wider context is needed for the classification task. We confirmed this hypotheses by classifying sentence-level embeddings obtained with BERT. Therefore, we performed the rest of our experiments on video-level data (derived from grouping sentence-level data), which is rich enough for tackling the topic detection task.

Our experimental results confirm that the textual information is rich enough for the task of topic detection, and can be solved, using text, with established NLP models such as BERT. On the other hand, we found that performing topic detection directly with keypoints is not as straightforward. We did not manage to solve it with this modality of data, and leave it as future work.

5.1. Modeling Setup

For this task, we dealt with two different kinds of data, namely, text and sign language represented as poses. For this reason, we have leveraged two different processing pipelines.

For dealing with sequences of poses, we chose the long-established LSTM [6] model, which is well-known for dealing with sequential data. As shown in Figure 9, we feed a pose to the LSTM cell at each time step, and use the hidden state h_T of the LSTM cell at the last time step T as input to a single-layer MLP that outputs the classification result.

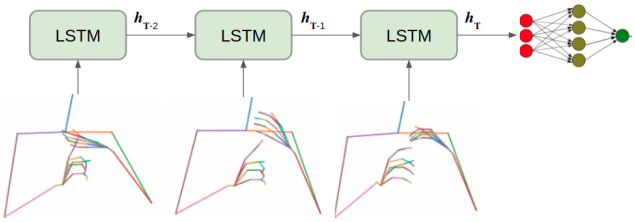

Figure 9. Topic detection setup for processing sign language sequences.

To deal with the textual data, we adopted a standard procedure in the domain of natural language processing. We feed BERT either a sequence of tokens (a sentence or a whole paragraph), and it outputs a rich representation in the form of a vector embedding. We then use this embedding as input to our multi-layer perceptron classifier [13] (MLP). The BERT feature extractor remains frozen with its pretrained weights, and only the classifier's weights are updated.

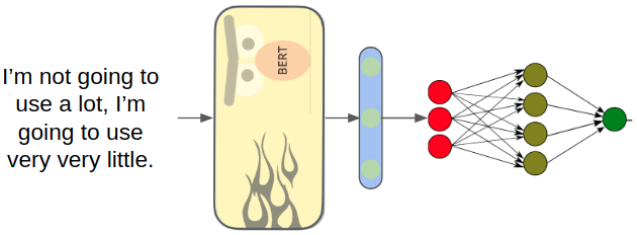

Figure 10. Schematic view of the embedding extraction and classification process for textual data. We extract textual embeddings from a pretrained transformer, BERT, which we then feed to a trainable MLP classifier.

5.2. Experiments

5.2.1 Classifying sentences

We begin by exploring whether sentence-level data (i.e. single sentences) contain enough information for solving a standard text classification task with transformers.

As we can see in Figure 11, sentence-level data does not carry enough information for accurately detecting its topic, even for a standard procedure with state-of-the-art transformers. Therefore, we concluded that we needed to aggregate data to video-level, *i.e.* to concatenate sentences belonging to the same video into a single text or sequence of poses.

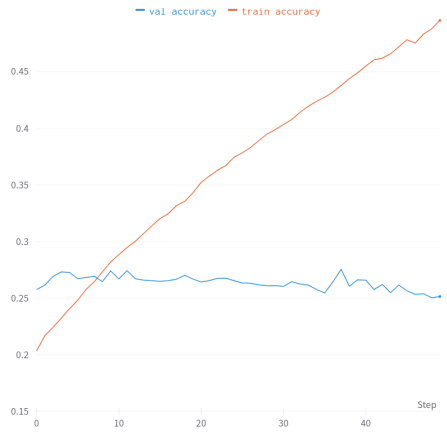

Figure 11. Accuracy curves on training and validation sets for classification on sentence-level textual data.

5.2.2 Classifying paragraphs

To make the topic classification task feasible, we aggregated data into paragraphs belonging to the same original video. In order to have a baseline against which to compare the results obtained by using sequences of poses as input to our model, we first tackled the task with textual data.

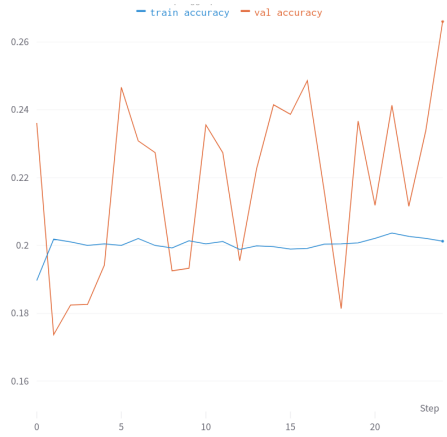

Figure 12. Accuracy curves obtained by using 2D sign language poses as input. In this case, the LSTM model struggles to learn any representations.

We performed hyperparameter tuning on the model described in Figure 10 to obtain the best possible accuracy on the validation set. The best combination resulted in a validation accuracy of 77%. This implies that the topic detection task on the How2Sign dataset is solvable, at least by using textual data as input.

Next, we asked ourselves whether 2D data is sufficiently rich to be used as inputs for solving the classification task, so we tried feeding 2D non-lifted poses to our model described in Figure 9.

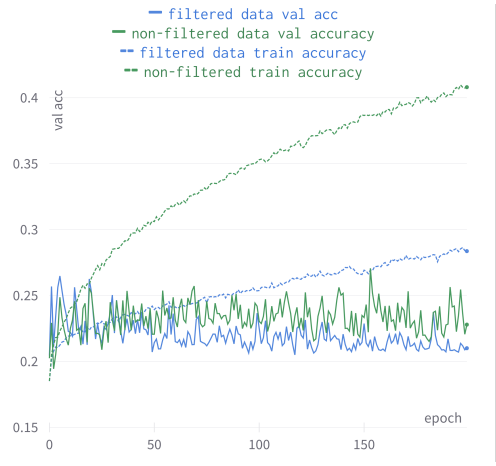

Figure 13. Overfit train, predict the majority class in val. Filtering prevents overfitting.

As shown in Figure 12, 2D keypoints are not suitable for sign-language processing, since they do not even allow for overfitting on the train set.

Finally, we tackled the R6D data, both non-filtered and filtered, since our initial goal with the topic detection task was to assess whether filtered data was better than non-filtered data.

Therefore, we trained our LSTM model with filtered poses in R6D, and with non-filtered poses also, in order to compare the model's performance depending on the data.

In Figure 13 we can see that, while post processed data achieves a slightly better accuracy on the validation set, there is no substantial improvement overall. The most noticeable difference is that the model tends to overfit the training set more easily with non-postprocessed data.

6. Discussion

Current hand pose estimation methods are often unreliable and do not always produce quality pose estimations. In this work, we address this issue by exploring a method for obtaining higher-quality hand pose estimations in the domain of sign language. We focus on the recently proposed Body2Hands, initially designed for conversational settings, and transfer it to the domain of sign language. We determine that it is not usable as an off-the-shelf method for our purpose. Moreover, we also indicate that it is not adequate for generating unseen sign language hand poses. Nevertheless, we show through a qualitative assessment that our method for hand-pose enhancement obtains promising results. Despite the fact that it does not yield improved results on the surrogate topic detection task, we believe a more powerful model architecture, different to an LSTM, such as transformers will be needed to properly tackle the topic detection task and thus obtain a fair assessment of our hand-pose enhancement method.

References

[1] Necati Cihan Camgoz, Simon Hadfield, Oscar Koller, Hermann Ney, and Richard Bowden. Neural sign language translation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7784–7793, 2018.

[2] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[4] Amanda Duarte, Shruti Palaskar, Lucas Ventura, Deepti Ghadiyaram, Kenneth DeHaan, Florian Metze, Jordi Torres, and Xavier Giro-i Nieto. How2Sign: A Large-scale Multimodal Dataset for Continuous American Sign Language. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[6] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 11 1997.

[7] Sang-Ki Ko, Chang Jo Kim, Hyedong Jung, and Choong Sang Cho. Neural sign language translation based on human keypoint estimation. *CoRR*, abs/1811.11436, 2018.

[8] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, Wan-Teh Chang, Wei Hua, Manfred Georg, and Matthias Grundmann. Mediapipe: A framework for building perception pipelines. *CoRR*, abs/1906.08172, 2019.

[9] Evonne Ng, Shiry Ginosar, Trevor Darrell, and Hanbyul Joo. Body2hands: Learning to infer 3d hands from conversational gesture body dynamics. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11865–11874, 2021.

[10] Rodrigues' rotation formula (wikipedia). `https://en.wikipedia.org/wiki/Rodrigues'_rotation_formula`.

[11] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021.

[12] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015.

[13] Frank Rosenblatt. Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms. In *Spartan Books, Washington DC*, 1961.

[14] Ramon Sanabria, Ozan Caglayan, Shruti Palaskar, Desmond Elliott, Loïc Barrault, Lucia Specia, and Florian Metze. How2: A large-scale dataset for multimodal language understanding. *CoRR*, abs/1811.00347, 2018.

[15] Ben Saunders, Necati Cihan Camgöz, and Richard Bowden. Progressive transformers for end-to-end sign language production. *CoRR*, abs/2004.14874, 2020.

[16] Donglai Xiang, Hanbyul Joo, and Yaser Sheikh. Monocular total capture: Posing face, body, and hands in the wild. *CoRR*, abs/1812.01598, 2018.

[17] Jan Zelinka and Jakub Kanis. Neural sign language synthesis: Words are our glosses. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3395–3403, 2020.

[18] Jan Zelinka, Jakub Kanis, and Petr Salajka. *NN-Based Czech Sign Language Synthesis*, pages 559–568. 07 2019.

[19] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. *CoRR*, abs/1812.07035, 2018.